EXPLORING THE USE OF LOCAL CONSISTENCY MEASURES AS THRESHOLDS FOR DEAD RECKONING UPDATE PACKET GENERATION

Dave Roberts*, Damien Marshall*°, Seamus McLoone°, Declan Delaney°, Tomas Ward°

** Centre for Virtual Environments*
Business House,
University of Salford,
Salford, M5 4WT
United Kingdom

d.j.roberts@salford.ac.uk, and
damienm@cs.nuim.ie

°National University of Ireland Maynooth
Maynooth,
Co. Kildare,
Ireland

seamus.mcloone@eeng.nuim.ie
decland@cs.nuim.ie
tomas.ward@eeng.nuim.ie

Abstract

Human-to-human interaction across distributed applications requires that sufficient consistency be maintained among participants in the face of network characteristics such as latency and limited bandwidth. Techniques and approaches for reducing bandwidth usage can reduce network delays, exploiting available bandwidth by reducing the network traffic. However, these approaches induce inconsistencies within the level of human perception. Dead reckoning is a well-known technique for reducing the number of update packets transmitted between participating nodes. It employs a distance threshold for deciding when to generate update packets. This paper questions the use of such a distance threshold in the context of absolute consistency and it highlights a major drawback with such a technique. An alternative threshold criterion based on time and distance is examined and it is compared to the distance only threshold. A drawback with this proposed technique is also identified and a hybrid threshold criterion is then proposed. However, the trade-off between spatial and temporal inconsistency remains.

1. Introduction

Communication technology allows collaborative activities across distributed teams, offering many advantages in today's globalised socio economic information culture. Although it is sometimes useful for individual members of a team to focus on distinct attributes of the activity, there must be some points of synchronisation. More generally collaborative activities require some level of consistency. However, current fundamental approaches to computer networking induce inconsistencies that are within the level of human perception. Consistency has been defined in different ways (Delaney 2005). In the context of this paper consistency is defined as state synchronisation of the local and remote users. If a state is synchronized in terms of position only it is spatially consistent; if it is synchronized in terms of time only, it is temporally consistent and if it is synchronized in both time and position it is

temporally-spatially consistent. The latter is also referred to as absolute consistency (Gautier et al. 1999).

The greatest single obstacle to achieving consistency is network latency, a phenomenon that can be attributed to a number of aspects such as queuing and processing at routers, bridges and gateways within the network, transcoding delays, propagation and transmission delays due to the speed of the communications link. In tandem with latency is the unpredictable variation in latency with time, which is referred to as jitter. In addition to latency and jitter, another obstacle to achieving consistency is the network bandwidth, which defines throughput of traffic. If the traffic exceeds the bandwidth then data will need to be buffered until flow decreases. This buffering can occur at the sending or receiving machine or at intermediate network switches, depending upon the greatest bottleneck at any given time. Thus delays will increase every time the bandwidth is exceeded. An objective of any distributed application must therefore be to minimize the end user's perception of inconsistency while maximizing exploitation of network resources.

Numerous techniques and approaches that optimize the use of network bandwidth and assist the maintenance of consistency have been documented. These can be subdivided into three general categories:

(1) Information Management Techniques: These all optimize the bandwidth usage by reducing the amount of information transmitted across the network. Examples include predictive contract agreement mechanisms, relevance filtering, packet bundling and packet compression;

(2) Time Management Techniques: These include two sub categories: those that manage consistency such as total ordering, causal ordering and wall clock synchronisation; and those that hide some perceivable effect such as delayed consistency, time warp and local perception filtering;

(3) Software and Hardware Architecture Techniques: These all aim to improve the efficiency of processing or disseminating information. Examples include QoS, protocols, network architectures and software design.

These techniques and approaches have been summarized and described by a number of authors (Joslin et al. 2004; Roberts 2004; Delaney 2005).

An important predictive contract agreement mechanism utilized by many distributed interactive applications is dead reckoning (IEEE 1993; IEEE 1995). This approach filters entity state changes to remove redundant information according to a distance threshold value. An entity is an element of the synthetic environment that is created and controlled by the simulation application. To generate update packets, the actual position of each local entity is compared to the position predicted by a parametric model of entity position. When the difference between the actual and predicted positions differs by the threshold value, an update packet is generated and transmitted to all other user computer nodes. These nodes then use the same parametric model to predict the entity's position until a new update is again received. A smaller threshold results in more frequent update packets and serves to increase temporal inconsistency.

This paper questions the use of a distance threshold as a suitable criterion for generating update packets when using dead reckoning. We propose an alternative criterion that is based on a measurement of both time and distance. Comparative analysis shows that the new criterion imposes an upper bound on absolute inconsistency, while the former method results in a wide range of unpredictable inconsistency values. However, the proposed technique also has its own drawback and a final hybrid solution is suggested.

The paper is structured as follows. Section 2 questions the use of a distance-based threshold in a dead reckoning predictive contract algorithm. A novel time-space threshold metric is then proposed. A comparative analysis of both threshold metrics is described in section 3. Section 4 highlights a drawback with the proposed metric and considers a the use of a hybrid metric. The paper ends with some concluding remarks in section 5.

2. The Dead Reckoning Threshold Problem

Prediction contract mechanisms, such as the standard Dead Reckoning (IEEE 1995) and the Hybrid Strategy Model (HSM) (Delaney et al. 2003; McCoy et al. 2005) techniques, reduce the number of update packets sent across a network, thereby reducing the effects of latency and increasing temporal consistency. However, this does not always translate into a corresponding increase in absolute consistency, which is a common misconception.

The successfulness of dead reckoning (and also HSM) is largely dependent on the threshold value employed. This value has a direct influence on the number of update packets generated and, therefore, on the temporal consistency between two nodes. However, the threshold value also determines the maximum modelling error that can occur between an entity's actual movement and its model of the movement. While each update packet synchronises the model and actual movement, the modelling error between updates can lead to poor spatial consistency. In essence, a large threshold value results in less update packets and, hence, a corresponding increase in temporal consistency, but also allows a larger local modelling error and, thus, an associated decrease in spatial consistency. The obvious question then arises – what is the impact of threshold on the absolute or temporal-spatial consistency? A delicate trade-off between temporal and spatial consistency is necessary.

This trade-off is clearly evident in the inconsistency metric proposed by (Zhou et al. 2003). Zhou et al defines the time-space inconsistency, Ω, as:

$$\Omega = \begin{cases} 0, & \text{if } |\Delta(t)| < \varepsilon \\ \int_{t_0}^{t_0+\tau} |\Delta(t)| dt, & \text{if } |\Delta(t)| \geq \varepsilon \end{cases} \quad (1)$$

where

Δ is the difference between the position of a local object and its remote replication;

$\Delta(t)$ is the above difference over a duration t;

t_o is the start time of the inconsistency and τ is the duration;

ε is the minimum perceivable distance.

Ω is the area under the graph of spatial consistency over time. When Ω = 0 absolute consistency has been achieved. Zhou et al. derive two equations to express inconsistency when dead reckoning is employed. The first of these, Ω_1, refers to the inconsistency accrued between the time an update packet is sent and the time it is received at the remote node:

$$\Omega_1 = \bar{v}\gamma T_d + \delta T_d + |a|_{\max} T_{DR} \frac{1}{2} T_d^2 \qquad (2)$$

where

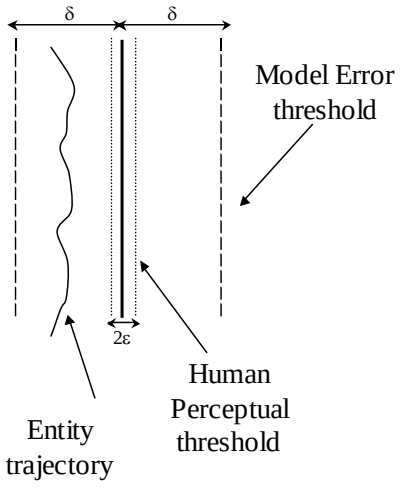
v is velocity;
γ is synchronisation between remote and local clocks;
T_{DR} is the time between updates;
δ is the dead reckoning threshold;
T_d is the time between an update being sent and received (latency);
A is acceleration.

The second, Ω_2, refers to the inconsistency accrued between the time an update packet is received at the remote node and the time the next update is sent by the local node:

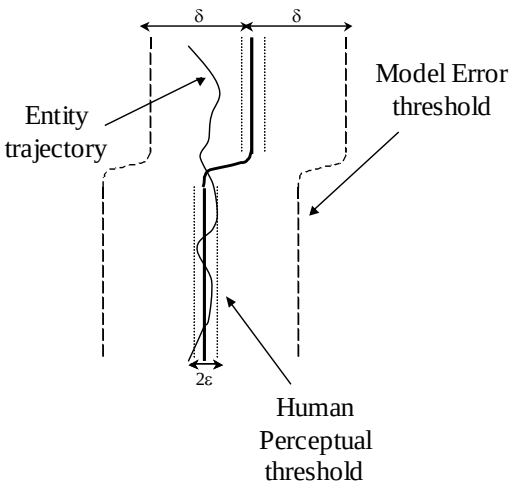
$$\Omega_2 = \left(\bar{v}\gamma + \frac{\delta}{2} \right) \left(T_{DR} - T_d \right) \qquad (3)$$

The final value of inconsistency is simply obtained by adding Ω_1 and Ω_2.

These equations clearly illustrate the duelling nature of temporal and spatial consistency in relation to the threshold value δ. Let us consider the simplified scenario where clock synchronisation, γ, is assumed to be zero and the velocity, v, is assumed to be constant. Now, as the threshold δ increases, the time between updates (T_{DR}) increases while the time between an update being sent and received (T_d) decreases. Thus, while decreasing T_d serves to reduce the absolute inconsistency value, increasing δ and T_{DR} only serves to increase this value. Once again, the question arises - what is the overall effect on absolute inconsistency?



(a)



(b)

Figure 1(a)-(b) – (a) An update is not generated as the entity remains within the spatial threshold; (b)

Evidently, using a distance-based threshold criterion in the dead reckoning prediction contract mechanism is questionable. Furthermore, another drawback exists with this threshold metric. Consider the diagram in Figure 1(a).

Should the local user remain within the distance threshold δ, but outside the perceivable error ε, than the absolute consistency increases indefinitely and will remain unbounded until such time as the user exceeds δ. Clearly, this is not an ideal scenario.

A Novel Threshold Metric

In an attempt to alleviate some of the drawbacks associated with the current technique, and inspired by Zhou et al's inconsistency metric, we propose sending update packets using a threshold value that is based on both distance and time.

Integrating the local modelling error over the time interval between two successive update packets results in a measure of local time-space inconsistency. Update packets are then generated when this local inconsistency value exceeds a specified threshold value. In this way the absolute consistency has an upper bound and will never be able to increase indefinitely, as shown in Figure 1(b) above.

The next section outlines the experimentation carried out to compare the use of a time-space threshold metric with that of a distance only metric, in a dead reckoning predictive contract algorithm. An analysis of the results is also given.

3. Experimentation and Analysis

The experimental test platform was implemented using the Torque game engine. The engine was extended to support logging of player position. Each simulation was carried out in a game environment that consisted of a unique route from a start to a target position. A plan view of the test environment is given in Figure 2 below.

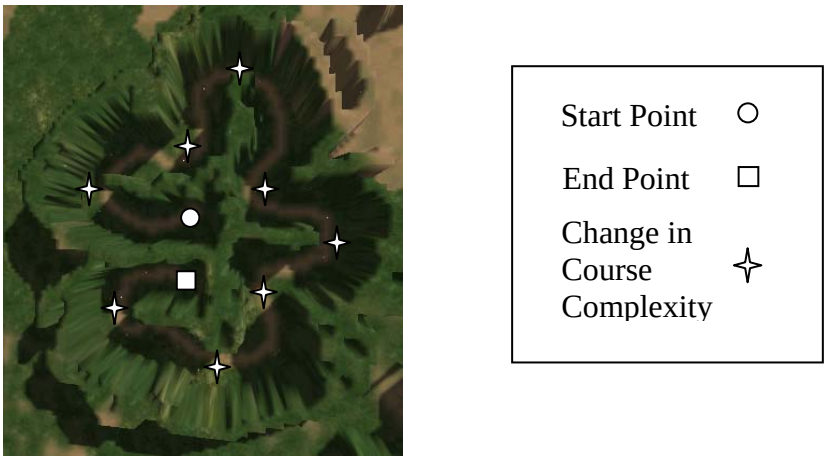


Figure 2 - Plan view of test environment

Positional data from three different participants of varying sex, age and virtual environment experience was collected. Each participant was first given a practice run with the environment in order to familiarise themselves with the controls and the environment itself. The behaviour of each participant was then recorded for three separate attempts at the course.

This data was then used as input for a Matlab simulation. This simulation applies a first order dead reckoning algorithm to the input data, which is recorded from the experimental platform. The threshold metric was varied for each experiment. The simulation results are now presented and discussed. It should be noted that results are given for only one player, but similar results were obtained for all players.

Figure 3 shows the actual player path, along with the modelled position using both the distance and the time-space threshold metrics. An error threshold of 7 game units was used for both models. The expanded view clearly shows that update packets are sent at difference times depending on the threshold metric used.

Figures 4(a) and 4(b) show the local inconsistency over time for a distance threshold metric and a time-space threshold metric respectively. Local inconsistency is determined by calculating the interval of the model error over time between updates. Each peak in both cases represents the time when an update packet is required. When this occurs, the measure of local inconsistency returns to 0.

Figure 4(a) shows a wide range of varying local inconsistencies when the distance threshold metric is used. In contrast, the space-time threshold metric results in a clearly upper-bounded local inconsistency measure, as illustrated in Figure 4(b), thus preventing an indefinite increase in inconsistency, as illustrated in Figure 1. This result seemingly vindicates the use of the latter threshold metric.

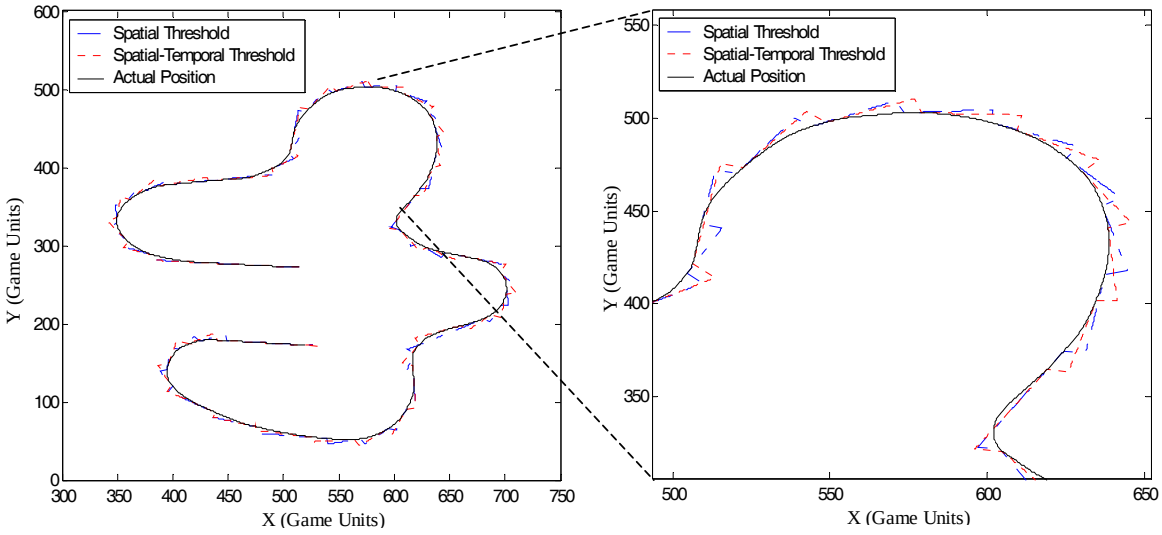
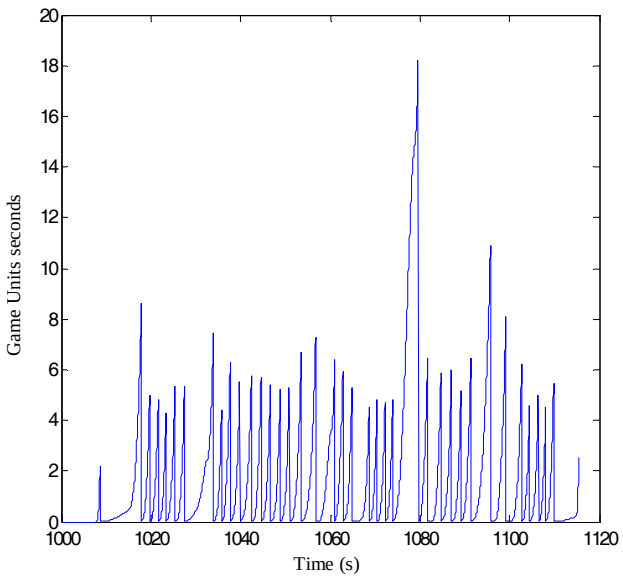
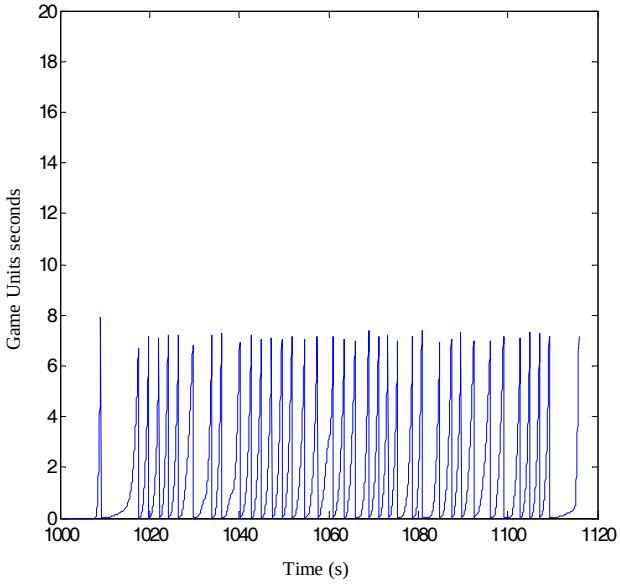


Figure 3 - Actual and modelled positions for an error threshold of 7. A zoomed-in section of the graph on the left is shown on the right for clarity.



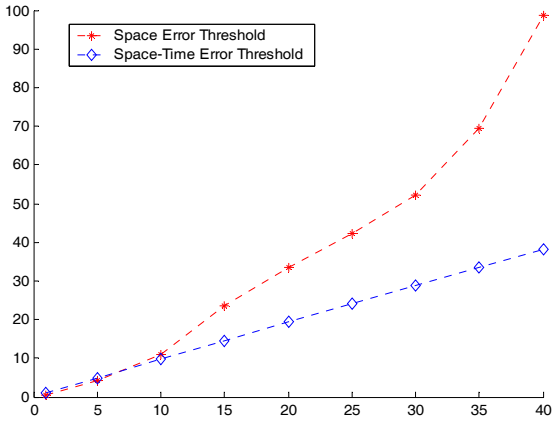
(a) Distance Threshold Metric



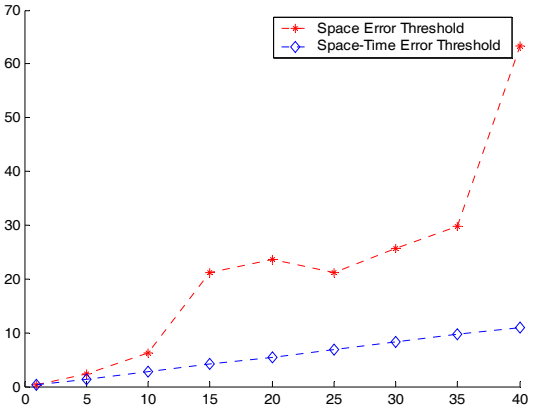
(b) Space-time Threshold Metric

Figure 4 (a)-(b) – Local inconsistency for an error threshold of 7

In Figures 3 and 4, a fixed error threshold of 7 game units was used. Figure 5(a) shows the average local inconsistency obtained for varying values of threshold. The corresponding standard deviation values are given in Figure 5 (b). These graphs show that as the specified error threshold value increases, so to does the average local inconsistency for both dead reckoning models. The model using the distance-based threshold results in a higher local inconsistency value in each case. Figure 5(b) confirms that the standard deviation of local inconsistency increases significantly with error threshold for the distance-based threshold model in comparison to its time-space threshold counterpart. Once again, these results highlight the key difference between both threshold metrics and confirm the results already obtained.



(a) average values



(b) standard deviation values

Figure 5 (a)-(b) – The average and standard deviation of local inconsistency for different values of threshold

4. Limitations of Proposed Technique

While the previous section presents results to support the use of a time-space threshold metric in a dead reckoning update packet mechanism, some limitations still exist.

Figure 6 shows how the average number of update packets required varies for different threshold values. Both threshold metrics are considered. While the proposed technique will affect the number of packets to be sent (more or less depending on the value of error threshold chosen), the trade-off between temporal and spatial consistency, as described in section 2, still exists. Hence the reduction in absolute inconsistency remains unclear.

In Figure 6, it is worth noting that at low threshold values, more update packets are required when the distance-based threshold metric is used. This is due to the fact that integrating the model error up to a point in time t_1 can actually result in a smaller value than the distance error between at time t_1. This highlights a further drawback with using the time-space threshold metric, as illustrated in Figure 7.

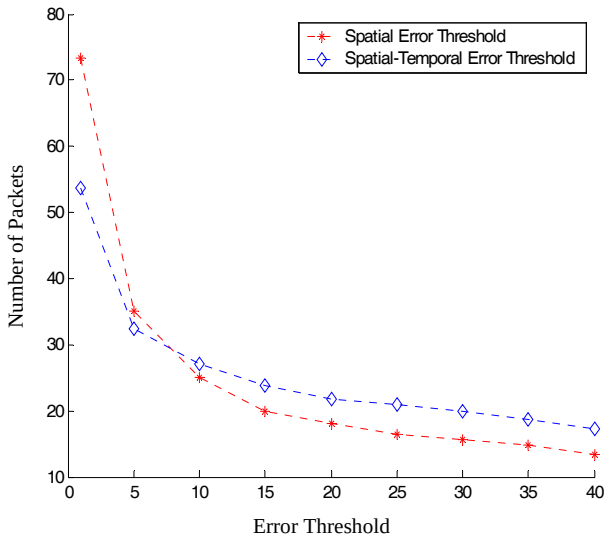


Figure 6 - Average packets required for increasing error threshold

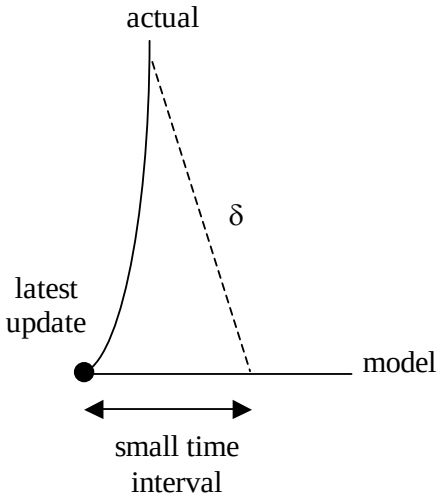
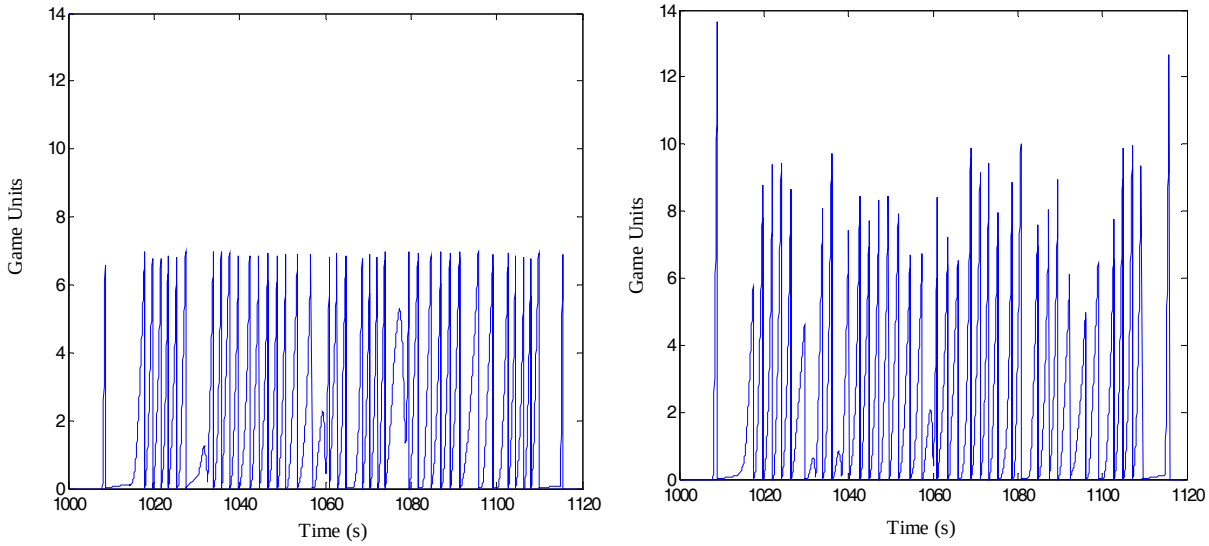


Figure 7 – Limitation of using a time-space threshold metric

In relation to Figure 7, consider the scenario where the dead reckoning model becomes very inaccurate very quickly. If a distance-based threshold metric is employed, then an update packet will be forthcoming almost immediately, which is what we would hope for. However, if a time-space threshold metric is used, it could take significantly longer before an update packet is sent, as a small time interval counters the large modelling error. This means that an entity's remote movement could potentially be very inaccurate over a short period of time, resulting in extremely poor spatial consistency. This is evident in the results shown in Figure 8 below. Note these results are in stark contrast to those in Figure 4.



(a) Distance Threshold Metric (b) Space-time Threshold Metric

Figure 8 – Local spatial inconsistency for an error threshold of 7

Further examination of Figures 4 and 8 suggest that the obvious solution is to use both threshold metrics in one dead reckoning algorithm. In other words, both metrics are simultaneously evaluated and as soon as one has reached the error threshold, an update packet is sent and both metrics are reset.

Thus, by combining both distance and time-space threshold metrics, the two situations outlined in Figures 1 and 7 can be prevented. Figure 9 shows the local inconsistency for a dead reckoning routine using the hybrid metric.

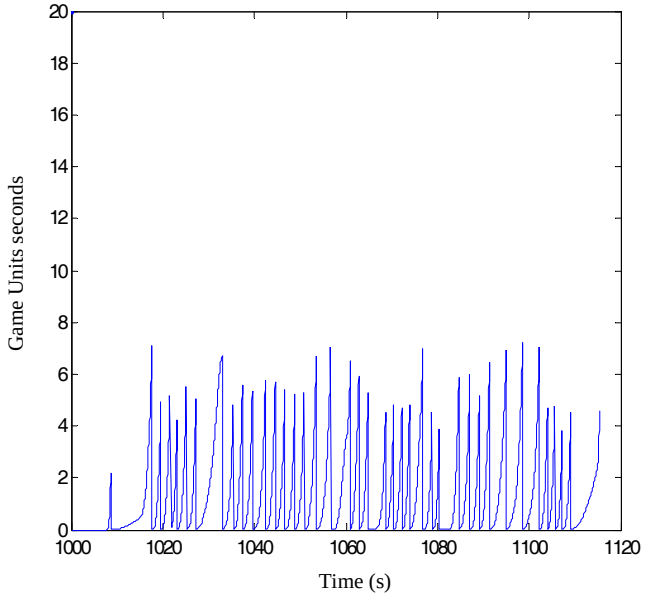


Figure 9 - Local inconsistency obtained, using the hybrid metric and error threshold of 7

The effect of simultaneously using the two threshold metrics is clearly evident when Figure 9 is compared to Figures 4 (a) and (b). The upper bound for local inconsistency in Figure 4 (b) is also present here. Furthermore, we no longer have to wait until the upper bound is reached before an update is sent, thus ensuring that the situation in Figure 7 no longer occurs.

4. Conclusion

This paper has examined the issue of consistency and the use of thresholds when employing an entity prediction packet reduction technique. In particular, the use of a distance threshold in dead reckoning was examined and it was noted that this can result in indefinite inconsistency. In order to combat this, a new time-space metric was proposed as a threshold for generating update packets. The use of this metric prevents the case of indefinite inconsistency but can result in a large spatial inconsistency over short time periods. A novel hybrid metric was then proposed that resolves both issues. However, in all three metrics, the trade-off between temporal and spatial consistency remains. This forms the basis of our future work, which will focus on understanding this trade-off and the value of absolute consistency.

Acknowledgements

This work is supported by Science Foundation Ireland and Enterprise Ireland under grant IRCSET/SC/04/CS0289.

References

Delaney, D., T. Ward and S. Mc Loone (2003). Reducing Update Packets in Distributed Interactive Applications using a Hybrid Model. 16th International Conference on Parallel and Distributed Computing Systems, Reno, USA, pp. 417-422.

Delaney, J. D. (2005). Latency Reduction in Distributed Interactive Applications using Hybrid Strategy-based Models. Electronic Engineering. PhD. Maynooth, National University of Ireland, Maynooth**:** 271.

Gautier, L., C. Diot and J. Kurose (1999). End-to-End Transmission Control Mechanisms for Multiparty Interactive Applications on the Internet. 18th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '99), New York, NY, USA, IEEE, pp. 1470-1479.

IEEE (1993). IEEE Standard for Distributed Interactive Simulation - Application Protocols. IEEE Std 1278-1993I. C. Society. New York, IEEE.

IEEE (1995). IEEE Standard for Distributed Interactive Simulation - Application Protocols. IEEE Std 1278.1-1995 (Revision of IEEE Std 1278-1993)I. C. Society. New York, IEEE.

Joslin, C., T. Di Giacomo and N. magnenat-Thalmann (2004). Collaborative Virtual Environments: From birth to Standardization. IEEE Communications Magazine. **44:** 28-33.

McCoy, A., S. McLoone, T. Ward and D. Delaney (2005). Dynamic Hybrid Strategy Models for Networked Multiplayer Games. 19TH European Simulation Multiconference (SCS-ESM 2005), Riga , Latvia, The Society for Modeling and Simulation International, pp. Accepted for Publication.

Roberts, D. J. (2004). Communication Infrastructures for Inhabited Information Spaces. Inhabited Information Spaces: Living With Your Data (Computer Supported Cooperative Work). D. N. Snowdon, E. F. Churchill and E. Frecon. London, Springer-Verlag**:** 233-267.

Zhou, S., W. Cai, B.-s. Lee and S. J. Turner (2003). "Time-space Consistency in Large Scale Distributed Virtual Environment." ACM Transactions on Modeling and Computer Simulation **14**(1): pp. 31-47.